

Trabajo de Laboratorio Número 2

Sistemas con Microprocesadores

16/9/2026

Instrucciones

- Los problemas de ejercitación propuestos en el presente trabajo de laboratorio pueden ser resueltos en forma individual o grupal.
- El planteo de la solución debe realizarse basándose en lo aprendido en las clases teóricas.
- Puede utilizar las clases de consulta para resolver problemas de interpretación de los enunciados y para verificar la validez de las soluciones obtenidas.
- Incluya comentarios en sus programas para que otros puedan entenderlo.
- Enfatique claridad, simplicidad y buena estructura. No pierda estos atributos para lograr mayor performance, a menos que así se especifique en el enunciado. Los programas deben ser razonablemente eficientes.
- Emplee rótulos y variables que posean significado, por ej. SUMA o PRUEBA en lugar de X o Y.
- En todos los casos comience por un diagrama de flujo de nivel intermedio (similar a un código escrito en lenguaje C).
- Recuerde que el procesador utilizado tiene un ordenamiento de byte del tipo Little-endian por lo cual la parte menos significativa de un número se ubica en la dirección de memoria mas baja.
- Posteriormente, se tomará una evaluación con problemas similares a los de este trabajo, la cual deberá ser resuelta en forma individual. La fecha de evaluación se encuentra en el sitio web de la materia.
- Las subrutinas transparentes deben cumplir el estándar ABI RISC-V.

Conceptos involucrados

- Programación avanzada de RISC-V.
- Subrutinas.
- Paso de Parámetros.

Problemas propuestos

- 1) Escriba una subrutina transparente que realice la suma de un número de 64 bits y uno de 32 bits. Esta subrutina recibe la dirección del número de 64 bits en **a0** y el número de 32 bits en **a1**. El resultado, de 64 bits, se almacena en el mismo lugar donde se recibió el primer operando. Escriba un programa principal para probar el funcionamiento de la misma, mostrando por pantalla el resultado final en HEXA. Por ejemplo:

Parámetros	a0 = 0x1008.0000	a1 = 0xA056.0102
Pre condiciones	M[a0] = 0x8100.0304	M[a0+4] = 0x0020.0605
Resultado	M[a0] = 0x2156.0406	M[a0+4] = 0x0020.0606

- 2) Implementar una subrutina transparente que realice el incremento de los segundos representados como dos dígitos BCD almacenados en dos direcciones consecutivas de memoria. La misma recibe el valor numérico 1 en el registro **a0** y la dirección de memoria donde está almacenado el dígito menos significativo en el registro **a1**. La subrutina devuelve en el registro **a0** el valor 1 si ocurre un desbordamiento de los segundos y se debe efectuar un incremento en los minutos, o 0 en cualquier otro caso.
- 3) *[Recomendado]* Implemente utilizando la subrutina desarrollada por usted para el ejercicio anterior, un programa principal que usando dos llamadas sucesivas permita incrementar los segundos y minutos. Utilice la llamada al sistema para mostrar por pantalla el valor de segundos y minutos cada 30 segundos hasta alcanzar los 10 minutos.
- 4) *[Recomendado]* Modifique la solución desarrollada en los últimos 2 ejercicios para que se pueda incrementar o decrementar el valor del par BCD almacenado en memoria. Para el caso de decremento se utilizará el valor -1 en el registro **a0**. Mostrar por pantalla cada 30 segundos una secuencia incremental de 5 minutos y a continuación una decremental de 10 minutos.

- 5) Escriba una subrutina que invierta una cadena de caracteres ASCII se encuentra almacenada en memoria. Esta subrutina recibe el puntero al inicio de la cadena en **a0** y un puntero al final en **a1**.
- Diseñe una subrutina **cambiar** que invierta el primer y el último carácter de la cadena. Esta subrutina recibe el puntero al primer elemento en **a0** y al último en **a1**.
 - Diseñe ahora una subrutina iterativa **invertirIT** que utiliza la rutina anterior para intercambiar los elementos de la cadena.
 - Diseñe ahora una subrutina recursiva **invertirRE** que utiliza la rutina del apartado a para intercambiar los elementos del extremo de la cadena y a si misma para invertir el resto.
 - Mostrar por pantalla los resultados de cada subrutina con 2 cadenas distintas.

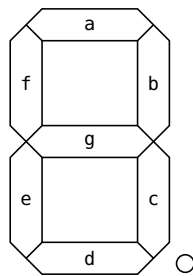
- 6) *[Profundización]* Se desea mostrar el valor de un número almacenado en memoria en su representación decimal en los displays de 7 segmentos.

- Escriba una subrutina transparente que reciba la dirección del número de 1 byte en **a0**. El mapa de bits de 7 segmentos de centena, decena y unidad del número convertido deberán almacenarse en las direcciones de memoria consecutivas a **a0**, respectivamente. En el siguiente ejemplo convertiremos el número 215 en su mapa de bits de 7 segmentos:

Pre condiciones $M[a0]=0xD7$ (215_{10})

Resultado $M[a0+1]=0x5B$ (2_{10}) $M[a0+2]=0x06$ (1_{10}) $M[a0+3]=0x6D$ (5_{10})

- Escriba un programa principal que llame a la subrutina del apartado anterior para verificar su funcionamiento.



b7	b6	b5	b4	b3	b2	b1	b0
	g	f	e	d	c	b	a

- Mostrar por pantalla el mapa de bits en HEXA, de centenas, decenas y unidades.